Amendments to the Specification:

Please amend paragraphs [0006] and [0028]-[0030] as follows:

[0006] Conventional methods for controlling the privileges granted to system resources include such methodologies as ~~Java's~~ JAVA's sand box model by Sun Microsystems of Palo Alto, California, Free BSD's "jail" function by the FreeBSD Foundation, and Linux's chroot "jail" function. None of these methodologies, however, can provide resource consumption controls flexible enough to be generically utilized within a grid environment. Consequently, a new methodology is required that reduces undesirable perturbations resulting from overhead.

[0028] In one embodiment, the ghost interface 210 can be implemented as one or more ~~Java~~ JAVA software objects. In such an embodiment, the ghost interface 210 can cause a ~~Java~~ JAVA Web server to be initialized with the ~~Java~~ JAVA debugging command, "java_g." The ghost interface 210 can utilize a ~~Java~~ JAVA debugging object to replicate the actions of the host 205 and convey the replicated actions 255 to the ghost agent 215. Additionally, passwords provided by the host 205 can be echoed to the ghost interface 210 and used to authorize the ghost agent 215 as appropriate.

[0029] In another embodiment that functions within a ~~Java~~ JAVA environment, both the host 205 and the ghost agent 215 can be implemented as different ~~Java~~ JAVA classes such that the ghost interface 210 can appropriately convey messages between the host 205 and ghost agent 215 classes. In yet another embodiment, the ghost interface 210 can be implemented using a Java/ Tcl blend, where Tcl is a computing language that interoperates with ~~Java~~ JAVA code segments. In that embodiment, the ghost interface

210 can use the "java::bind" command to generate callback scripts from events in the host 205. The call back scripts can replicate actions for the ghost agent 215.

[0030] The embodiments of the ghost interface 210 disclosed herein are not restricted to the ~~Java~~ JAVA programming language as one of ordinary skill in the art can utilize any of a variety of programming languages and techniques. For example, the ghost interface 210 can be implemented using a GNU debugger distributed by the Free Software Foundation and an Apache server distributed by the Apache Software Foundation. The GNU debugger can be attached to an Apache server causing all activity occurring within the server to be directed to the GNU debugger. The host 205 can be disposed within the Apache server so that the ghost agent 215 can utilize replicated actions of the host 205 provided by the GNU debugger.